Computer Vision for Volleyball Serve Classification

Chris Lee

Department of Computer Science, The College of Wooster

April 19th, 2024

Contents

1 Abstract **3**

2 Introduction **4**

2.1 Technology Advancement in Sports 4

2.1.1 In Major Professional Sports Leagues 4

2.1.2 In Volleyball 5

3 Background **5**

3.1 Machine Learning 6

3.2 Random Forest Algorithms 7

3.3 Deep learning and Convolutional Neural Networks 8

3.4 Computer Vision for Volleyball 10

3.4.1 Court Detection 11

3.4.2 Pedestrian Detection, Color Classification, Player Tracking, and Mapping 11

4 Serve Classification Project **13**

4.1 Software Description 13

4.2 Data Gathering and Cleaning 14

4.3 AlphaPose Implementation 15

4.4 Random Forest Algorithm Training for Classification 15

4.5 Conclusion and Results 16

5 Acknowledgement **17**

References **19**

1 Abstract

This project implements a novel solution for classifying volleyball serves using computer vision and machine learning techniques. Python, AlphaPose, and a scikit-learn Random Forest Algorithm are utilized to process and categorize short clips of volleyball serves as either float or topspin serves. Background is provided on the techniques and software used throughout our project. This project was written in a Google Colab and Drive environment. Clips of serves are gathered, standardized, and cleaned. They are then processed through AlphaPose to extract keypoints (e.g. shoulders, hips, elbows, wrists), which we use to calculate a backswing angle in each frame for each player's serve. We process the serves into two empty lists that contain sequences of backswing angles and labels for the serve type, then train a Random Forest Algorithm to classify serves based on this input. Our program successfully processes and classifies the test serve given (float1) which is not included in the training dataset.

2 Introduction

2.1 Technology Advancement in Sports

Sports are an integral part of many of our lives. Whether commiserating over a Super Bowl loss, competing in a volleyball tournament, or just playing pickup, these experiences with sports serve as a crucial avenue for community bonding — especially given the dearth of walkable cities and third spaces in states like Ohio. Sports leagues also have a significant impact economically from the local to international levels. Sports leagues benefit local economies by generating jobs and consumer spending, and with sports such as American Football generating 18.6 billion per year as an industry and growing, these local benefits exist at scale [5].

As the interest, engagement, and market capitalization of professional sports leagues such as the NFL and NBA has skyrocketed, so too has the attention given towards the training athletes receive. Teams regularly make significant financial investments in cutting edge research and technology. Technological advancement, largely in video analytics, has benefited training techniques across the board. Sports medicine, film study, and coaching schemes have all evolved in tandem with strides in medical science, computer vision, and statistical analysis.

2.1.1 In Major Professional Sports Leagues

In the NFL, film study is a cornerstone of game preparation. Coaches and players analyze game footage to understand opponent tendencies, strategize plays, and improve individual and team performance. Computer vision plays a crucial role in this process, allowing for automated player tracking, play recognition, and statistical analysis. Computer vision can also benefit teams and players in more subtle ways, such as injury prevention. Modern NFL uniforms are equipped with RFID sensors and receivers allowing the league to track and analyze player movement. The league uses this information in conjunction with high-resolution, high-frame-rate cameras to gain insights into how different plays and movements impact player safety[4]. Similar to the NFL, the NBA and MLS both rely heavily on film study for game analysis and player development. Computer vision technologies enable coaches and

analysts to track player movements, identify offensive and defensive patterns, and provide valuable insights into optimizing game strategies.

2.1.2 In Volleyball

While not as widely adopted as in major professional sports leagues, both collegiate and professional volleyball teams have started using computer vision for film study. Systems have been developed to track player movements, analyze ball trajectories, and provide performance feedback to players and coaches. A stellar example of this has been the startup Balltime, which within a few years has expanded into a host of teams from youth clubs to top collegiate teams such as Long Beach State[1].

This project aims to build on existing technological advancements in player training. We will implement a machine-learning based computer vision application to identify different kinds of volleyball serves. Our hope is this application will allow experienced players to analyze and tweak their serve performance without a coach, through providing descriptive statistics on their technique.

3 Background

Artificial Intelligence (AI), according to Google's Cloud Division, is "a field of science concerned with building computers and machines that can reason, learn, and act" in ways that normally "require human intelligence" or use "data whose scale exceeds what humans can analyze" [8]. AI will fundamentally reshape the way we experience life and engage with the global economy. According to the managing director of the IMF, AI will affect "40 percent of jobs around the world", and "60 percent of jobs" in advanced economies such as the United States [7]. We even find our day-to-day lives increasingly infiltrated, from Siri in our cell phones to ChatGBT in our professors' nightmares. The use of artificial intelligence both mirrors human intelligence and scales itself to datasets and problems beyond regular human capacity, including but not limited to the sports world.

3.1 Machine Learning

A key aspect of artificial intelligence is its ability to self learn. The growth in AI development in recent years has largely been driven by the subfield of machine learning. The most basic definition of machine learning is as "the field of study that gives computers the ability to learn without explicitly being programmed," from pioneer Arthur Samuel in the 1950's [2]. Machine learning employs algorithms to analyze input data and predicts outputs within acceptable bounds. These algorithms incrementally improve over time by learning from new data.

There are two main methods for training machine learning algorithms: supervised and unsupervised learning. They are primarily differentiated by their training data and learning objectives. Supervised learning trains models on labeled datasets, mapping input data to desired outputs and allowing for future predictions. Unsupervised learning trains models on unlabeled datasets, allowing for more free flowing pattern and structure recognition [6].

Supervised learning tackles problems where you have ground rules, or labels, for your data. Machine learning models that are supervised largely divide into two categories: regression

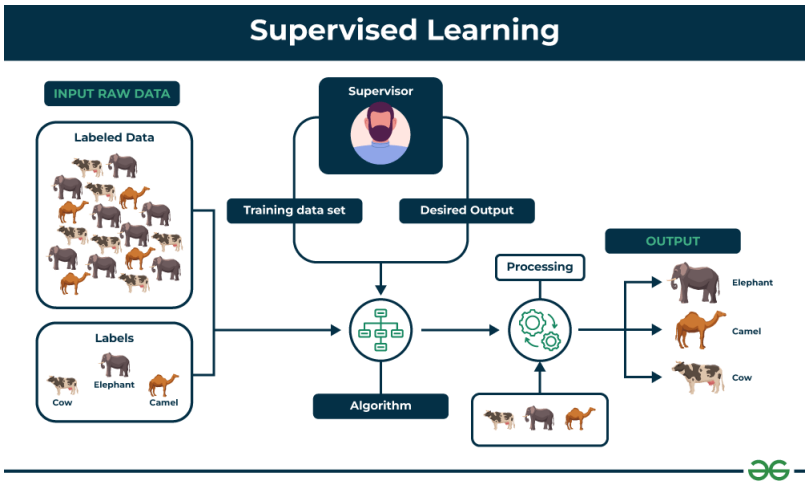

Figure 1: GeeksforGeeks: Supervised Learning

and classification. Regression is for when the labels are continuous to create a mapping function, while classification is when the labels are discrete and output is classified. Examples of regression include linear regression, polynomial regression, gradient boosting regression,

neural networks, and support vector regression. Examples of classification include decision trees, support vector machines, naive baye, and neural networks. Random forests can be utilized for both regression and classification tasks.

Since unsupervised models work with unlabeled data or data without explicit output labels, they are not typically categorized as regression or classification algorithms. Unsupervised models instead use k-means clustering, hierarchical clustering, and principal component analysis for tasks including clustering, association, dimensionality reduction, and pattern discovery. These models can be more difficult to gauge the accuracy of, but excel at uncovering

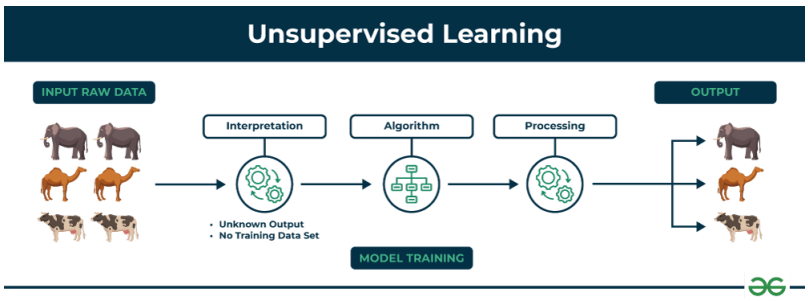

Figure 2: GeeksforGeeks: Unsupervised Learning

patterns, grouping data points, and reducing the complexity of high-dimensional datasets without relying on explicit guidance [6].

3.2 Random Forest Algorithms

The supervised classification model in our project used to classify serves from pose-estimation output is a random forest algorithm. Random Forests are a type of decision tree algorithm, representing models in a tree structure where each node signifies a feature and each branch a decision, leading to an output value at a leaf [12]. The process starts from the root node, traversing through the tree based on input values until a leaf with the result is reached.

Similar to Neural Networks, Random Forests are built via a learning process using training data. Here, trees are created step by step, emphasizing the importance of features in the application context. Unlike traditional Decision Trees, Random Forests address overfitting by constructing multiple trees from different random subsets of training data and features at each node, shown in figure 3 on page 8. The ensemble of trees mitigates errors through

majority voting for classification or averaging for regression, enhancing model robustness. Random Forests exhibit resilience to missing data as not all trees utilize the same features, ensuring accurate predictions even with missing values. However, they may struggle with regression problems involving target values outside the range of training data. Despite this limitation, Random Forests typically achieve high-quality results and are commonly utilized with tree sizes ranging from 100 to 500 [12].

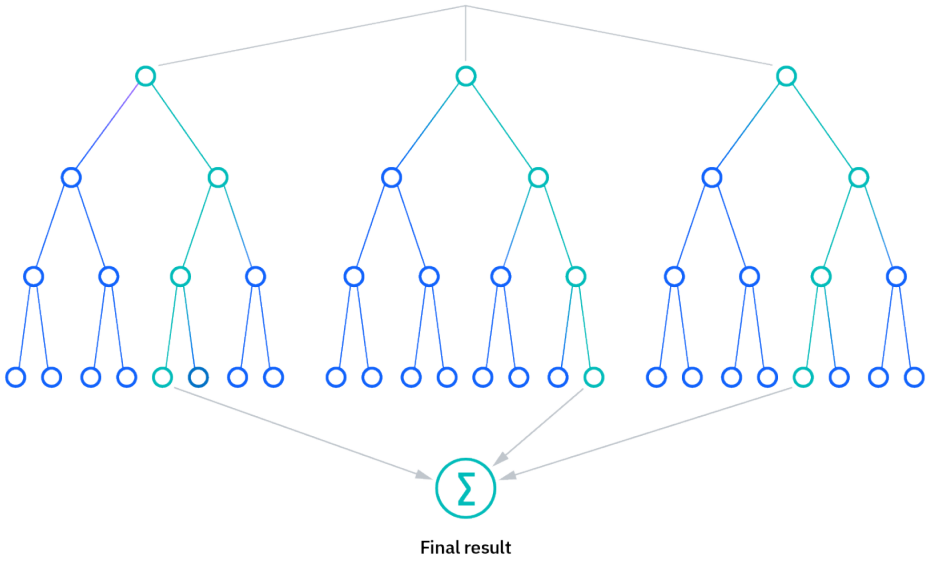

Figure 3: IBM: Decision Tree Ensemble in a Random Forest Algorithm

3.3 Deep learning and Convolutional Neural Networks

Most neural networks belong to a subsection of machine learning known as deep learning. Figure 4 on page 10) illustrates how deep learning is nested within machine learning, artificial intelligence, and algorithms as a whole. Deep learning is a subset of machine learning "that uses multi-layered neural networks" composed of artificial neurons nodes to simulate the processing function of the human brain [9].

Neural Networks comprise input, hidden, and output layers, transforming input data into usable output. They handle numeric data, passing through input to hidden layers, then scaled to desired output. Forward propagation and back propagation form the functional core of Neural Networks. Forward propagation is the predictive step in which each hidden or

output node receives inputs from previous nodes, which are multiplied by weights, summed, and transformed using a function before passing to the next layer or to the output[12]. This occurs for each hidden and non-hidden layer before output. Backpropagation is the training step in which the network takes the final output loss and recursively calculates the loss function to optimize parameters. The network's architecture, including layer count, neuron count, and transformation functions, is set before training. Deepening a Neural Network essentially involves adding more hidden layers to increase increase complexity and process hierarchical features.

Convolutional Neural Network (CNN) is a neural network model optimized for processing data with grid patterns, such as images, and has adaptive learning properties that lend well towards pattern analysis. In their work "An Introduction to Convolutional Neural Networks", O'Shea and Nash outline that a CNN's basic functionality can be summarized into four key areas:

1. "As found in other forms of ANN, the input layer will hold the pixel values of the image.

2. The convolutional layer will determine the output of neurons of which are connected to local regions of the input through the calculation of the scalar product between their weights and the region connected to the input volume. The rectified linear unit (commonly shortened to ReLu) aims to apply an 'elementwise' activation function such as sigmoid to the output of the activation produced by the previous layer.

3. The pooling layer will then simply perform downsampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation.

4. The fully-connected layers will then perform the same duties found in standard ANNs and attempt to produce class scores from the activations, to be used for classification. It is also suggested that ReLu may be used between these layers, as to improve performance." [11]

While these areas provide a clear methodology difference between CNNs and standard neural networks, a difference in data processing is the usage of three dimensional neurons, which include height, width, and depth. Depth refers to the third dimension of an activation volume, not the total number of layers. Neurons in a layer connect to only a small region of the preceding layer. For example, an input 'volume' might have dimensions of 64x64x3, resulting in a final output layer of 1x1xn, where n is the number of classes, condensing the input into class scores across the depth dimension [11].

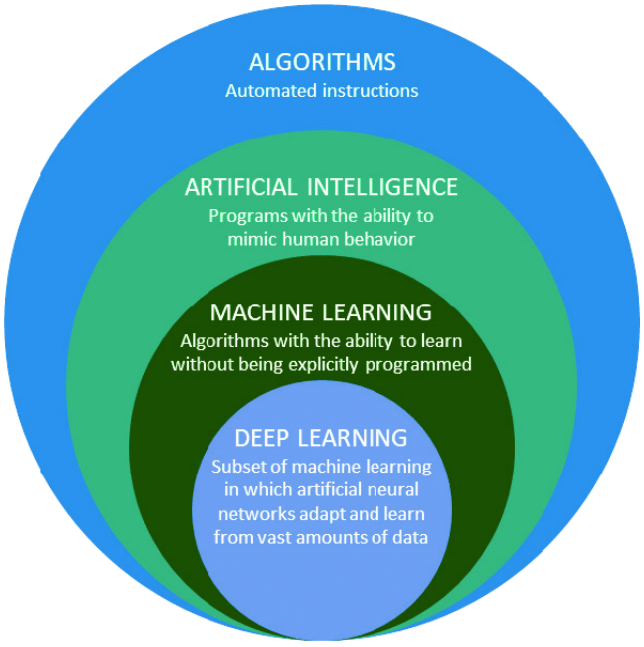

Figure 4: Johannes Vrana: Visualization of algorithms vs. artificial intelligence vs. machine learning vs. deep learning

3.4 Computer Vision for Volleyball

Computer vision is, as the name suggests, a field of artificial intelligence dedicated to enabling computers to derive visual information. It involves developing algorithms and techniques to extract meaningful insights from images or videos. This encompasses a wide range of tasks, such as object detection, image classification, facial recognition, and scene understanding. Computer Vision in particular has seen some of the greatest advances in importance and usage in sports with the growth of sports analytics — with plenty of room for future innova-

tion. Convolutional Neural Networks and derivative neural networks have become dominant in computer vision.

Implementing computer vision involves a number of iterative steps. The work "Player Tracking and Analysis of Basketball Plays," by Cheshire, Halasz, and Perin provides a relatively strong template for 2D image processing with object detection, which is most relevant for analyzing single angle videos of volleyball serves. This project used position tracking and 2D court reconstruction to represent basketball players and their on-court movements based off clips pulled from YouTube, identifying five total steps of: Court Detection, Pedestrian Detection, Color Classification, Player Tracking, and Mapping [3]. While not all 5 steps are relevant to our project, this approach provides a solid framework to tackle computer vision analysis for multi-athlete sports.

3.4.1 Court Detection

Timothy Lee's project "Automating NFL Film Study: Using Computer Vision to Analyze All-22 NFL Film", while using similar methods as other sports video analysis papers, is differentiated by the extraction of relevant information about football plays using only recorded video from one camera. [10] This is useful as many beach volleyball clips, even at the highest level, may only cover one or two camera angles (and usually never simultaneously).

Timothy Lee used vertical yardlines and horizontal hashmarks from the NFL field to identify and subsequently reconstruct the field from one shot (see figure 1 on page 12). This reconstruction utilized a Hough Transform, a feature extraction technique used to convert images from Cartesian to polar coordinates in order to identify complex shapes "such as as lines, circles and ellipses." Given the scope of our model which used the hitter's backswing angle to classify serves, court detection is somewhat tangential to this project's implementation. However a robust court detection system would allow recognition of foot faults, as well as create avenues for deeper insight into how the arc and direction of the toss impacts a serve.

3.4.2 Pedestrian Detection, Color Classification, Player Tracking, and Mapping

Object detection is often the most important component and the biggest challenge, with a variety of environmental factors impacting the algorithms efficacy. Each sport has unique

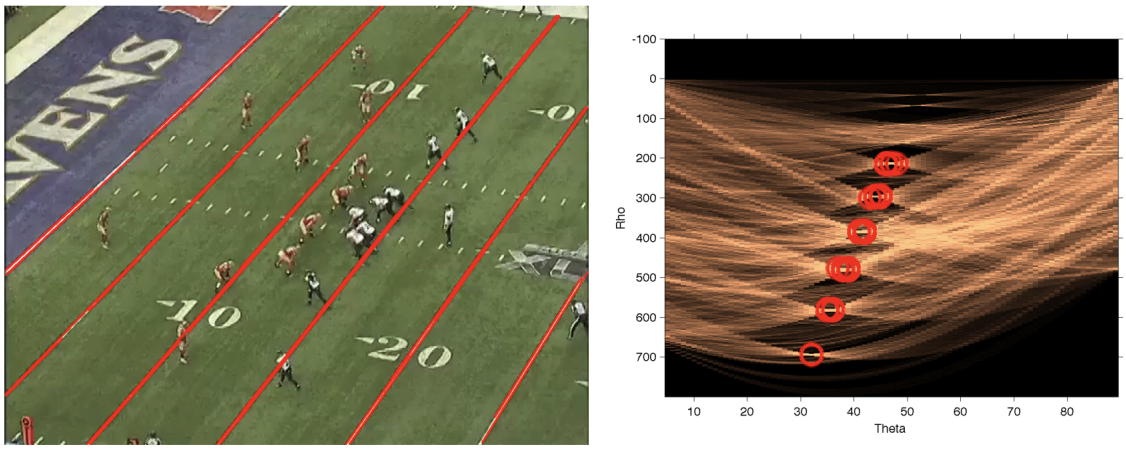

Figure 5: Timothy Lee: Automatic detection of yardlines on a football field (left) using the Hough transform (right).

challenges and as such requires tweaked models. Most sports-based papers referenced in this project, including Cheshire et al. and Timothy Lee, work hand in hand with color classification for object detection to identify teams. This approach using color classification is largely inapplicable for analyzing a single clip of a volleyball server.

For the sake of this project, we aim to identify a server and the different joints of their body — including hips, shoulders, elbows, and wrists as the primary components to identify types of serves. Previous literature exists examining serve type detection based on ball movements and trajectories. [13] However, no publicly available research has covered serve type detection based on player movement itself. Effective player movement analysis has positive implications not only for serve types but also to provide feedback on body mechanics, training, and player safety. As such, the program is built around the detailed identification of a single server with little consideration to differentiating them from other players not included in the shot. Our approach to achieve this task would be to utilize Alphapose, a pre-trained pose detection model based on the R-CNN (Region-based Convolutional Neural Network) architecture. Unlike many other pose detection models or CNN models, AlphaPose uses a bottom-up approach by using keypoint coordinates to first identify individual body parts, then estimate the pose. This is ideal for our project as we generate backswing angles by using the keypoint coordinates of joints most important to the swing.

4 Serve Classification Project

4.1 Software Description

Now that we have covered the background necessary including machine learning, convolutional neural networks, and decisions trees, we can discuss our implementation of the volleyball serve classification project. We use Python, AlphaPose, and a scikit-learn Random Forest algorithm to process and classify short clips of volleyball serves.

For our development workspace, this project utilizes integration with Google Colab and Drive. Using Azure with Jupyter notebook, GitHub Codespaces, or Bridge to Kubernetes in VS Code was considered for each IDE individually. We ultimately settled on the google environment given the relatively small scale of our project, use of free computing credits, and ease of integration with google drive. This made it easier to keep track of, modify, and share our dataset of volleyball clips.

A variety of external libraries are used to ensure smooth integration between all parts of the project. These include the following:

- PyDrive: For accessing and manipulating files using python on Google Drive. We use this for downloading pre-trained models, uploading processed results, and parsing through those results.

- torch and torchvision: Deep learning libraries particularly applicable for PyTorch-based neural network modeling. We use this to load our pre-trained neural network models and perform inference using AlphaPose

- pyyaml: A YAML parser and emitter for Python. Since YAML is a superset of JSON, this allows us to output our data from AlphaPose into a JSON format and parse using YAML, and is broadly useful to configure parameters for various components.

- scipy: A library used for scientific computing and technical computing, providing user-friendly and efficient numerical routines.

- pillow: A Python Imaging Library (PIL) fork, used for opening, manipulating, and saving many different image file formats. We use this in preprocessing and postprocessing

for our video frames.

- cython_bbox: A library for fast bounding box operations written in Cython. We use this to process the bounding box coordinates obtained from AlphaPose.
- numpy: A package for scientific computing with Python and ML algorithms, providing support for large, multi-dimensional arrays and matrices along with a collection of mathematical functions to operate on these arrays.
- scikit-learn: A machine learning library in Python, providing simple and efficient tools for data mining and data analysis. We use this to train and evaluate our Random Forest Algorithm used for serve classification of our processed pose-estimation output data.
- keras: A neural networks API that acts as an interface for the TensorFlow library. We use sequence padding from this library to standardize our array size for the random forest input data.
- json: A library for handling the JSON data format. We use this for loading pose estimation results obtained from AlphaPose.
- os: A module in Python for operating system-dependent functionality. We use this for navigating file directories, creating directories, and executing shell commands.
- math: A module providing mathematical functions. We use this to calculate the backswing angle using keypoints coordinates from our AlphaPose output.

4.2 Data Gathering and Cleaning

We sought to gather clear clips of two different types of volleyball serves for this project: topspin and float. We use YouTube to search for pro-level jump serves that show correct form. We then download them as mp4s and collate them into two equal size sets of serves. Since our project uses a classification machine learning model in AlphaPose with raw video data, we want to clean the input video data as much as possible. Video standardization

is done manually with DaVinci Resolve. Cleaning input video data helps ensure pose-estimation is only applied to the server and prevents extraneous errors in our classification model. Each video is clipped to 1 second, from the final step to completion of the swing. Additionally, each video is cropped to avoid erroneous identification of other players/pedestrians.

4.3 AlphaPose Implementation

In the AlphaPose implementation step, we set up and leverage the AlphaPose model to extract keypoints (e.g. shoulders, hips, elbows, wrists) from the input video data. This output can be seen visually and in JSON format in our sample_vids subfolders.

Next, we prepare the JSON output data for processing, pathing to, loading, and verifying the integrity of the files. Using the math library, we build a backswing angle calculation with our calculate_angle function. We determined the backswing angle as an ideal training input instead of raw keypoints since keypoints coordinates are not standardized across videos. This is due to the varying placement of the servers in each video and variations in each servers technique (such as excess loading in the penultimate jump). Keypoint indices are defined based on COCO ordering from AlphaPose documentation.

We then initialize empty arrays sequences[] and labels[] to store sequences of backswing angles and labels for serve type. For each topspin serve's JSON file we iterate through each frame, identifying keypoint coordinates, calculating the backswing angle, and appending the angle to the serve sequence. When the serve is completely processed, we append the serve sequence to the sequences array, and append a label "0" to the label array for topspin. Figure 6 on page 16 illustrates the code for this process. We repeat for the float serves with an equivalent "1" label for float. Finally, we standardize the serves using the pad_sequences function from Keras.

4.4 Random Forest Algorithm Training for Classification

Our last step is to train a random forest algorithm to classify serves. A variety of models were tested including a support vector machine, a simple neural network, and a linear regression.

```
# Initialize empty lists to store backswing angles and labels
sequences = []
labels = []

# Iterate through each topspin serve JSON entry
for serve in topserves:
    serve_sequence = []
    for frame_data in serve:
        keypoints = frame_data["keypoints"]

        shoulder = keypoints[shoulder_idx * 3 : shoulder_idx * 3 + 2]
        hip = keypoints[hip_idx * 3 : hip_idx * 3 + 2]
        wrist = keypoints[wrist_idx * 3 : wrist_idx * 3 + 2]
        # keypoint coordinates

        backswing_angle = calculate_angle(shoulder, hip, wrist)
        # backswing shoulder angle

        serve_sequence.append(backswing_angle)
        # appending to serve sequence

    sequences.append(serve_sequence)
    # appending sequence to list

    labels.append(0)
    # appending topspin label
```

Figure 6: Alphapose Output Processing for Random Forest Algorithm

The random forest algorithm had the highest accuracy out of all our tests and models, and consistently correctly classified serves correctly. We theorize the random forest algorithm worked best for classification due to our small dataset. As discussed in the background, random forest algorithms can be trained more accurately and efficiently with a relatively small amount of data. Our final output, Figure 7 on page 16, shows that the random forest algorithm correctly classifies the our test serve not included in the training dataset (float1).

```
# Predict
prediction = rf_classifier.predict(new_padded_sequence)
if prediction == 0:
    print("The serve is classified as a 'top' serve.")
else:
    print("The serve is classified as a 'float' serve.")
```

```
The serve is classified as a 'float' serve.
```

Figure 7: Random Forest Algorithm Classification of Test Serve "float1"

4.5 Conclusion and Results

Overall, our project resulted in the accurate classification of serves into float and topspin categories. After applying the AlphaPose pose-estimation model on each video, we parsed

the JSON output files to gain keypoint coordinates, then used these coordinates to calculate a backswing angle. Data standardization techniques were applied throughout the gathering, cleaning, and output processing of keypoint coordinates. Backswing angles were fed into a sequences array (with a sequence of backswing angles for each serve), and a corresponding label of "0" for top or "1" for float was given in the labels array for each sequence. After padding, we successfully trained a random forest algorithm on the dataset.

Challenges to validity for this project included the small size of our dataset and limitations of a random forest algorithm for classification. A larger dataset, simply put, is necessary for a robust classification project. While our program successfully classified serves in all our tests, it's important to note that all our videos were highly standardized including isolation of the server, and our test set was relatively limited. The accuracy metric of 1.0 given by the scikit accuracy_score function suggests our dataset was too small. While random forest algorithms are ideal for small datasets such as ours, they result in diminishing returns in accuracy for larger datasets. Unfortunately, we were limited by time and by the GPU usage set by Google Colab, reducing the robustness of our analysis. Future work includes expanding the dataset, utilizing a more scalable classification model, and integrating additional aspects of the serve such as the arc and direction of the ball.

5 Acknowledgement

It is my deepest pleasure to thank Professor Daniel Cohen-Cobos who guided me (indirectly) all the way from Long Beach State to complete this writeup based on his template. I would also like to extend my gratitude to The College of Wooster Computer Science Department for its support in my research endeavor.

Finally, thank you to Dr. Guarnera for helping me clarify my ideas and always offering amazing advice throughout the course of this term.

References

[1] AI Platform for Volleyball Highlights & Analytics. Balltime, 2024. URL: `https://www.`

balltime.com/.

[2] Sara Brown. Machine learning, Explained. Ideas Made to Matter, April 2021. URL: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained.

[3] Evan Cheshire, Cibele Halasz, and Jose Krause Perin. Player Tracking and Analysis of Basketball Plays. Stanford University, EE368/CS232 Digital Image Processing, Spring 2015. URL: https://web.stanford.edu/class/ee368/Project_Spring_1415/Posters/Cheshire_Halasz_Perin.pdf.

[4] CIO. How AI Is Helping the NFL Improve Player Safety. CIO, February 9 2024. Accessed March 9, 2024. URL: https://www.cio.com/article/1306736/how-ai-is-helping-the-nfl-improve-player-safety.html.

[5] Forbes. Total revenue of all National Football League (NFL) teams from 2001 to 2022 (in billion U.S. dollars) [Graph]. In Statista, August 2023. Retrieved April 19, 2024. URL: https://www.statista.com/statistics/193457/total-league-revenue-of-the-nfl-since-2005/.

[6] GeeksforGeeks. Supervised and Unsupervised Learning. GeeksforGeeks, March 2024. Last Updated: 13 Mar, 2024. URL: https://www.geeksforgeeks.org/supervised-unsupervised-learning/.

[7] Kristalina Georgieva. AI Will Transform the Global Economy. Let's Make Sure It Benefits Humanity. IMF Blog, 2024. URL: https://www.imf.org/en/Blogs/Articles/2024/01/14/ai-will-transform-the-global-economy-lets-make-sure-it-benefits-humanity.

[8] Google Cloud. What is Artificial Intelligence (AI)? Google Cloud, 2024. URL: https://cloud.google.com/learn/what-is-artificial-intelligence.

[9] IBM. What is deep learning? IBM, Year the page was last accessed. URL: https://www.ibm.com/topics/deep-learning.

[10] Timothy E. Lee. Automating NFL Film Study: Using Computer Vision to Analyze All-22 NFL Film. CS231A Final Project Progress Report, Stanford University, 2024. URL: https://web.stanford.edu/class/cs231a/prev_projects_2015/LeeTimothy.pdf.

[11] Keiron O'Shea and Ryan Nash. An Introduction to Convolutional Neural Networks. 2015. Department of Computer Science, Aberystwyth University, Ceredigion; School of Computing and Communications, Lancaster University, Lancashire. URL: https://arxiv.org/pdf/1511.08458.pdf.

[12] Peter Roßbach. Neural Networks vs. Random Forests – Does it always have to be Deep Learning? Frankfurt School Blog, 2018. URL: https://blog.frankfurt-school.de/wp-content/uploads/2018/10/Neural-Networks-vs-Random-Forests.pdf.

[13] Constantin Toporov. Volleyball Serve Detection With Machine Learning. *Better Programming*, 2023. URL: https://betterprogramming.pub/volleyball-serve-detection-with-machine-learning-68288a7e68eb.